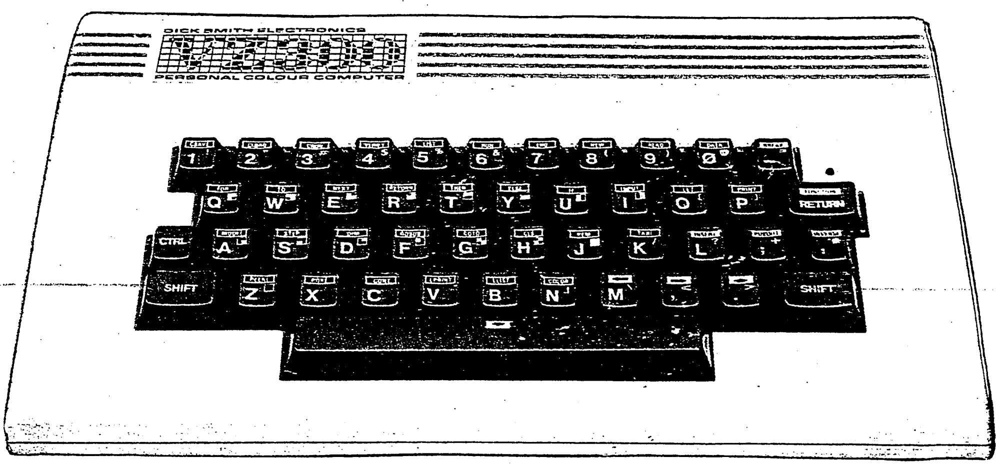
SEPT/OCT 1987 ISSUE # 14

HUNTER VALLEY VZ JOURNAL



PRODUCED BI-MONTHLY

BY HUNTER VALLEY VZ USER'S GROUP

STOP PRESS :- ** NEW ** NEW ** NEW **
Larry TAYLOR has just updated his VZ EPSON PRINTER PATCH V1.4 with host of extra features. Review in next issue. It's a must for anyone with an EPSON or EPSON compatible printer. Available from VSOFTWAREZ.

POTPOURI :- Dot Matrix Addendum - VZ User Groups and Publications. Page 3

EDUCATIONAL PROGRAM by Paul LEON :- Page 4
With last school term just around the corner students will find this program helpfull in learning their Elements and Symbols.

CHARACTER CODES (C) by Robert QUINN :- Pages 5-7
Once again Robert comes to the fore to demistify the VZ, this time it's the characters and their codes.

MAILING LIST UPDATE :- Page 7
Dick SMITH'S Technical Bulletin 111 on tape MAILLIST to disk SAVE/LOAD conversion gave just the bare essentials. The mods on page 7 give MAILLIST users a bit more versatilty. Major mods in next issue.

AEM4505 SPEECH SYNTH. PROGRAMS by Dave BOYCE :- Pages 10
Held over for a couple of issues they finally make an appearance.

INKEY$ INPUT ROUTINE by Paul LEON :- Page 10
Unlike the INPUT command an INKEY$ INPUT routine can be tailored to your requirements. In the M/WORDS program in this issue nearly all inputs are handled by this routine.

USING DISK TOKENS by Robert QUINN :- Pages 11-13
Carrying on from previous article Robert has supplied us with a M/L shift routine to use with disk tokens as a demonstration program. It's a very fast BLOCK MOVE routine. For another example of it's use see M/WORDS program.

32K BIB RAM by Joe LEON :- Page 13
Most comments regarding the 2K and 8K BIB RAMS from previous issues also apply to this unit. It's too expensive for me.

8K BIB RAM PART II by Joe LEON :- Pages 14-16
Last issue had the circuits and now the program for activating any missing word you like and more. Because of it's built in LITHIUM batteries which should last around ten years you can think of it as a MAGIC EPROM.

VZ TOKENS AND WORDS by Robert QUINN :- Pages 17-18
I'm afraid the GREMLINS (See BELIEVE IT OR NOT) got in last issue with some ERRORS on page 11 and all the TOKENS were wrong in the right column on page 13. The complete word and token tables are reproduced which also includes all disk words as well. Intending constructors of the 8K BIB RAM will find the information most useful.

FOR SALE - FOR SALE :- Our usual ads appear once more pages 19-20

BELIEVE IT OR NOT :-
(Gremlins) - Normally I would apologise, but as my son Paul claims he tought me all about computers I'll let him take the blame.

CONGRATULATIONS SHIRLEY AND DAVE BOYCE ON YOUR NEW ARRIVAL BABY GIRL ---- JANET ELIZABETH BOYCE

PS :- IF YOUR NAME IS DAVE THEN WATCH OUT AS YOU COULD BE NEXT -
- YOU CAN'T SAY YOU HAVE'NT BEEN WARNED -

DISK DRIVE PRIMER - - - - - - -

Apparently there are a few VZ users with DISK DRIVES who find some of the terms used confusing, mainly about file types.

T:MAILLIST - The 'T' before the Filename denotes a TEXT or BASIC program. You would either RUN"MAILLIST" or LOAD"MAILLIST" .

D:MAILDATA - The 'D' denotes a DATA file and can only be created by a program like MAILLIST. All the Names, Addresses, etc, you typed in are contained in a 'D' type DATA file and The commands used to SAVE or LOAD your DATA file are :- OPEN, CLOSE, PR# and IN#. The 4 commands can only be used from within a program, exept for CLOSE which can be used in direct mode.

FOR PRIVATE SALE - - - - - - -

1 off EXTENDED BASIC - 'XB' as from LASERLINK - $25.00
1 off LIGHT PEN again from LASERLINK - $32.00
Both items are as new and used twice only and still in original packing. Ring Dave on (08) 384 6574 about both items.

DOT MATRIX PRINTERS ADDENDUM

My thanks to John D'Alton for sending down the fonts below which arrived too late for Larry Taylors Dot Matrix article in last issue. They were printed on a CITIZEN 120-D printer which John markets.

REVERSED PRINT

DOUBLE HEIGHT AND EXPANDED

VZ USER GROUPS AND PUBLICATIONS

VZ USER	MARK HARWOOD	P.O.BOX 154 DURAL N.S.W. 2158
LE'VZ OOP	J.C.E. D'ALTON	39 AGNES St. TOOWONG QLND. 4066
VZ DOWN UNDER	SCOTT LE BRUN	5 CAMERON COURT WANTIRNA 3152
VZ-LINK NEWSLETTER	P.O.BOX 1972	C.P.O. AUCKLAND NEW ZEALAND
VZ USERS GROUP	P.O.BOX 22-094	CHRISTCHURCH NEW ZEALAND

HUNTER VALLEY VZ USERS' GROUP - P.O. BOX 161 JESMOND N.S.W. 2299
EDITOR-JOE LEON (049)51 2756 - SECRETARY-ROSS WOODS (049)71 2843

SUBSCRIPTION - H.V.VZ.JOURNAL - 6 MONTHS $9.00 - 12 MONTHS $18.00
New Zealand - 6 MONTHS $12.00 - 12 MONTHS $24.00

MEETINGS - 2nd. FRIDAY of MONTH at NEW LAMBTON COMMUNITY CENTRE CNR. ALMA RD. and CROMWELL ST. NEW LAMBTON - Upstairs Hall.

NOTE :- When writing to any above or H.V.VZ. Users' Group for information please enclose a S.S.A.E. or 2 Int. Reply Coupons.

The program below is designed to help learn ELEMENTS and their SYMBOLS. You can chose ELEMENTS, SYMBOLS or RANDOM of the two previous. Whichever you chose, you'll have to type in either the ELEMENT's name or it's SYMBOL. Even if you know the element's name the spelling has to be 100% correct. Have fun.

Only 20 elements have been included and you can add or replace with different elements. Do not forget to reDIMension if you add or subtract number of elements.

```
10 '*********************************************************
20 '*   TRAINING PROGRAM FOR LEARNING ELEMENTS AND SYMBOLS   *
30 '*   TOGETHER WITH PERCENT RIGHT SCORE --- BY PAUL LEON   *
40 '*********************************************************
50 :
60 POKE30744,96:DIM E$(20),S$(20):GOSUB100:GOSUB700:FORI=1TO20
70 N=RND(20):IFE$(N)<>""THEN70
80 READE$(N),S$(N):NEXTI:GOTO200
90 :
100 CLS:PRINT TAB(8)" ELEMENTS QUIZ "
110 PRINT" =============================="
120 PRINT"  NUMBER OF QUESTIONS ANSWERED"
130 PRINT"  RIGHT >          WRONG >"
140 PRINT"  PERCENTAGE CORRECT -->"
150 PRINT" ==============================":RETURN
160 :
200 PRINT@226,"QUESTION #      WAS":PRINT@418,"QUESTION #"
210 FORI=1TO20:PRINT@429,USING" ##";I:GOSUB600
220 PRINT@482," PRESS Q TO QUIT ";
230 IFEL$="E"THEN310ELSEIFEL$="S"THEN250
240 A=RND(2)-1:IF A THEN310
250 PRINT@290," ELEMENT      ";E$(I)
260 PRINT@354,;:INPUT" SYMBOL     ";A$
270 IFA$="Q"THENGOSUB500:GOSUB600:GOTO370
280 IFA$=S$(I)THENGOSUB500ELSEIFA$<>S$(I)THENGOSUB510
300 NEXT:GOSUB600:GOTO370
310 PRINT@354," SYMBOL       ";S$(I)
320 PRINT@290,;:INPUT" ELEMENT    ";A$
330 IFA$="Q"THENGOSUB500:GOSUB600:GOTO370
340 IFA$=E$(I)THENGOSUB500ELSEIFA$<>E$(I)THENGOSUB510
360 NEXT:GOSUB600
370 PRINT@440,"Y     ":PRINT@418,;:INPUT" LIKE TO TRY AGAIN  ";T$
380 IF T$="Y"THENRUNELSEIFT$="Q"THENCLS:END:ELSE370
390 :
500 PRINT@247," RIGHT ":SOUND31,1:R=R+1:GOTO520
510 PRINT@247," WRONG ":SOUND15,4:W=W+1
520 PRINT@108,USING"##";R:PRINT@124,USING"##";W
530 PRINT@152,USING"###.##";R/I*100
540 PRINT@237,USING" ##";I:RETURN
590 :
600 FOR J=288TO388STEP32
610 PRINT@J,"                                ";:REM 32 SPACES
620 NEXT:RETURN
690 :
700 PRINT@290,"ELEMENTS - SYMBOLS - RANDOM":PRINT@379,"R"
710 PRINT@355,;:INPUT"SELECT  E OR S OR R   ";EL$
720 IFEL$<>"E"ANDEL$<>"S"ANDEL$<>"R"THEN700ELSERETURN
790 :
800 DATA HYDROGEN,H,HELIUM,HE,LITHIUM,LI,BERYLLIUM,BE
810 DATA BORON,B,CARBON,C,NITROGEN,N,OXYGEN,O,FLOURINE,F
820 DATA NEON,NE,SODIUM,NA,MAGNESIUM,MG,ALUMINIUM,AL,SILICON,SI
830 DATA PHOSPHOROUS,P,SULFUR,S,CHLORINE,CL,ARGON,AR
840 DATA POTASSIUM,K,CALCIUM,CA
```

CODE :-

The program CODE gives you access to all the VZ character codes--ASCII, INVERSE, PEEK/POKE--from the keyboard. Each time you press a key (by itself or with SHIFT key held down) the corresponding character will display on screen along with its various codes in decimal and hex.

For nongraphic characters, the first two lines display the normal and inverse CHR$ codes. These are the codes used in PRINT statements. Normal CHR$ codes are ASCII codes with the range 32 to 95, as set out in the BASIC REFERENCE MANUAL. These divide into two sets : those from 32 to 63, which I will, for convenience, call numeric characters (digits, operation characters and punctuation characters) and those from 64 to 95, the so-called alphabetical characters.

For numeric characters, the inverse CHR$ code is the normal CHR$ code + 192.

For alphabetical characters, the inverse CHR$ code is the normal CHR$ code + 128.

Hold SHIFT and press X key to change the background of the screen from light to dark and vice versa. Visually, what is normal and what is inverse will depend on the background, but the CHR$ codes do not change.

The remaining two lines display the normal and inverse PEEK/POKE codes. These are the codes that determine what character will appear at a given position on the screen (in video memory) when you POKE a number to that position, or the number you get if you PEEK at a given position on the screen (in video memory).

PEEK/POKE codes are somewhat more complicated than CHR$ codes. Again, what is normal and what is inverse depends visually (what you see) on the background, but also the PEEK/POKE codes change according to the background.

With a dark background, the PEEK/POKE codes for normal alphanumeric characters range from 0 to 63, with the alphabetical characters having codes from 0 to 31 and numeric characters having codes from 32 to 63. Then the PEEK/POKE code for an inverse character is the normal PEEK/POKE code + 64 (range 64 to 127).

With a light background it is the inverse alphanumeric characters that have PEEK/POKE codes in the range 0 to 63 and the normal characters whose PEEK/POKE codes are the inverse PEEK/POKE codes + 64 (range 64 to 127). You can see this when you switch background (SHIFT X) while CODE is running.

Now try some of the graphic character keys (SHIFT J, SHIFT Z, etc.). CODE displays each graphic character in all eight colors, with the PEEK/POKE character code for each color. The PEEK/POKE code for each successive color (down the screen) is the previous color code + 16.

Green is the start color, and it is the green PEEK/POKE code for each graphic character that is the CHR$ code for that character.

The range of CHR$ codes for the sixteen graphic characters is 128 to 143. This is the standard or default range. There are three other ranges of CHR$ codes available for graphic characters: 144 to 159; 160 to 175; and 176 to 191. These simply repeat the standard range of graphic characters.

When using CHR$ codes for graphic characters, you select the color you want for a graphic character by using the COLOR command. With PEEK/POKE you select the color of a graphic character by choosing the PEEK/POKE graphic character code appropriate to the color. The PEEK/POKE code for a green graphic character is also the standard CHR$ code for that character. This is the code that CODE displays at top of screen.

CONTROL CODES :- these are CHR$ codes for cursor control commands that are accessed with the CTRL key held down: cursor up, down, left, right, INSERT, RUBOUT. RETURN key also has a CHR$ code.

Other CHR$ codes can be viewed with CODE by HOLDING SHIFT and pressing C key.

The BACK ARROW character (ASCII code 95; inverse code 223) is not accessible via the keyboard. You have to use CHR$ or POKE to make use of this character. With CODE running, SHIFT V keys will display the back arrow character codes.

```
10 '*****************************************************
12 '*  ASCII AND PEEK/POKE CODES FOR NORMAL - INVERSE   *
14 '*  GRAPHIC AND CONTROL CHARACTERS BY ROBERT QUINN   *
16 '*****************************************************
20 :
30 CLS:N$="0123456789ABCDEF":X=-1:VM=28672:POKE30744,1
40 PRINT@7,"CHARACTER CODES"
50 PRINT:PRINT"PRESS A KEY BY ITSELF"
60 PRINT:PRINT"OR WITH SHIFT HELD DOWN"
70 PRINT:PRINT"OR WITH CTRL FOR CURSOR CONTROL KEYS"
80 PRINT:PRINT"SHIFT X SWITCHES BACKGROUND"
90 PRINT:PRINT"SHIFT C FOR MISCELLANEOUS CODES"
95 PRINT:PRINT"SHIFT V FOR BACK ARROW"
100 A$=INKEY$:A$=INKEY$
110 IFPEEK(26875)=249,SOUND20,1:X=NOTX:POKE30744,ABS(X):GOTO190
120 IFPEEK(26875)=243THENSOUND20,1:CLS:GOSUB410:A$=""
125 IFPEEK(26875)=219THENSOUND20,1:A$=CHR$(95):GOTO140
130 IFA$=""THEN100
140 B=0:CLS:A=ASC(A$):B$=A$:IFA<31ORA=127THENGOSUB310
150 IFA>31ANDA<64THENB=A+192
160 IFA>31ANDA<64THENB=A+192:N=A:V=A+64
170 IFA>63ANDA<96THENB=A+128:N=A-64:V=A
180 IFA>127ANDA<144THENB=1
190 IFB>1THEN220ELSEIFB=0THEN100
200 T=0:FORR=1TO8:M=A+16*(R-1):COLORR:PRINT@T,B$"  ";M;:GOSUB500
210 T=T+64:NEXT:GOTO100
220 PRINT@64,B$;:PRINT@68,USING"###";A;:M=A:GOSUB500
230 IFA>31ANDA<96THENPRINT"    ASCII "
240 PRINT@128,CHR$(B);:PRINT@132,USING"###";B;:M=B:GOSUB500
250 PRINT"    INVERSE "
260 POKE256+VM,N:PRINT@260,USING"###";N;:M=N:GOSUB500
270 PRINT"    PEEK/POKE"
280 POKE320+VM,V:PRINT@324,USING"###";V;:M=V:GOSUB500
290 PRINT"    PEEK/POKE"
300 GOTO100
```

CHARACTER CODES CONT. 7

```
310 M=A:IFA=127THENPRINT@64,"RUBOUT   ";A;
320 IFA=21THENPRINT@64,"INSERT  ";A;
330 IFA=8THENPRINT@64,"CURSOR LEFT  ";A;
340 IFA=9THENPRINT@64,"CURSOR RIGHT ";A;
350 IFA=10THENPRINT@64,"CURSOR DOWN  ";A;
360 IFA=27THENPRINT@64,"CURSOR UP    ";A;
370 IFA=13THENPRINT@64,"RETURN       ";A;
380 GOSUB500:RETURN
410 PRINT@64,"CURSOR HOME  ";28;"= HEX 1C":PRINT
420 PRINT"CLEAR SCREEN ";31;"= HEX 1F"
430 RETURN
500 C%=M/16:M=M-16*C%:GOSUB520:C%=M:GOSUB520
510 PRINT" = HEX ";D$;:D$="":RETURN
520 D$=D$+MID$(N$,C%+1,1):RETURN
```

MAILING LIST UPDATE . . . .

This update is for those persons who converted their tape MAILING LIST for disk use according to D.Smith's Technical Bulletin 111 which is available free from D.Smith.

DISPLAY DISK DIRECTORY :- Just press '0' for DIRECTORY.

READ MENU :-
1) Selecting READ will load your DATA file from disk.
2) CLOSE OPEN DATA FILE. If you get FILE ALREADY OPEN ERROR, then type in GOTO1000, select 2. READ option and select 2 again to CLOSE OPEN FILE.

WRITE MENU :-
1) WRITE NEW DATA FILE - Use this option only if your disk has'nt a data file on it already called MAILDATA.
2) UPDATE OLD DATA FILE - This option will first ERASE MAILDATA file and then WRITE a new MAILDATA file.

WARNING :- In case of ERRORS - DO NOT type in RUN or you will lose all your data, instead type in GOTO1000 to regain control.

```
600 COLOR7:PRINT@34,"[illegible]"
605 PRINT@292,"1. READ DATA FROM DISK"
610 PRINT@356,"2. CLOSE OPEN DATA FILE"
630 PRINT@450,"[illegible]";:K$=INKEY$
640 D1$=INKEY$:IFD1$="2"THEN6135
650 IFD1$=" "THEN1000ELSEIFD1$="1"THENGOSUB30:RETURNELSE630
690 :
700 COLOR7:PRINT@34,"[illegible]"
710 PRINT@292,"1. WRITE  NEW DATA FILE"
720 PRINT@356,"2. UPDATE OLD DATA FILE"
730 PRINT@450,"[illegible]";:K$=INKEY$
740 D2$=INKEY$:IFD2$="1"ORD2$="2"THENGOSUB30:RETURN
750 IFD2$=" "THEN1000ELSE730
800 CLS:DIR:STATUS:PRINT:GOSUB4100:GOSUB40000:GOSUB30:RETURN
1020 COLOR7:PRINT@34,"[illegible]";:PRINT@45,;"# OF RECORDS:";
1025 PRINTUSING" ###";DT
1027 PRINT@98,"0. DISPLAY DISK DIRECTORY";
1230 IFI$<"0"ORI$>"9"THEN1220ELSESOUND30,1
1340 IFI$="0"THENGOSUB800:GOTO1020
5000 REM"[illegible]
5010 GOSUB30:COLOR7:GOSUB700
5020 PRINT@261,"[ WRITE DATA TO DISK ]":GOSUB4100:GOSUB30
5205 IFD2$="1"THEN5210ELSE:ERA"MAILDATA"
6000 REM"[illegible]
6010 GOSUB30:COLOR7:GOSUB600
6020 PRINT@260,"[ READ DATA FROM DISK ]";:GOSUB4100:GOSUB30
```

AEM'S CTS/SPO Speech Board - Program notes for the VZ 200/300

You will notice that the OUTPUT Routine for TALK A program is MUCH SHORTER and COMPACT when compared to my Original Programs.

In the routine 'TALK 3' the actual OUTPUT Routine takes a few lines, this is for Clarity Purposes.

In 'TALK A' you will see that the Output Routine has been reduced to just two lines.

Explanation of Output Routine taken from 'TALK 3'

10200 FOR T=1 TO LEN(A$) ' Starts Output Loop of A$

10220 A=ASC(MID$(A$,T,1)) ' A is made equal to the ASCII value of each letter in the String in turn

10240 OUT 12,A ' the ASCII value is output to the Printer.

10260 NEXT ' goes back for the next character in the String

10270 LPRINT:LPRINT:LPRINT ' forces the Speech to Output ALL of the String

OUTPUT Routine from 'TALK A'

1500 is the combination of lines 10200 to 10260 as above.
1520 is the same as 10270 above.
I have found that only 2 LPRINT commands are needed.

Dave BOYCE

```
50 GOTO 100
60 ERA"TALK 3"
70 SAVE"TALK 3":DIR:STATUS
80 END
100 CLEAR 300 :' FILE - TALK 3
180 CLS:PRINT"TALKER VERSION 1.3"
200 PRINT
10110 ' SIGN ON
10120 A$="BY--YOR--COMMAND":GOTO 10180
10130 'INPUT SECTION
10140 PRINT" SEPERATE WORDS WITH A DASH"
10150 PRINT" E.G.->  HELLO-THERE-FRIEND":PRINT
10160 INPUT" TALK PT.1)";A$:INPUT" TALK PT.2)";B$:A$=A$+"-"+B$
10170 IF A$="-" THEN A$="PLEEZ-ENTER-SUM-THING-2-SAY"
10180 '
10190 ' OUTPUT LOOP ROUTINE
10200 FOR T=1 TO LEN(A$)
10220 A=ASC(MID$(A$,T,1))
10240 OUT 12,A
10260 NEXT
10270 LPRINT:LPRINT:PRINT
10280 GOTO 10140
10290 ' GO BACK TO SENDER
10300 END
```

```
10 CLS ' FILE - TALK A
15 ' DO NOT EDIT LINE 210
20 POKE31946,161
25 PRINT"   CTS/SPO SOUND EFFECTS DEMO.":PRINT :'FOR AEM 4505
30 CLEAR 300
35 PRINT"       1) GARGLING"
40 PRINT"       2) RUBBER LIPS"
45 PRINT"       3) TRAIN"
50 PRINT"       4) SNEEZES"
55 PRINT"       5) MACHINE GUN"
60 PRINT"       6) WHISPERING"
65 PRINT"       7) STARTER MOTOR"
70 PRINT"       8) SIREN (SORT OF)"
75 PRINT"       9) DOG"
80 PRINT"      10) OWN SOUND/MESSAGE"
85 PRINT"      11) UP-DATE DISK"
90 PRINT
200 PRINT@416,;:INPUT" ENTER YOUR CHOICE ";A
210 & A GOTO 500,550,600,650,700,750,800,850,900,950,1650
220 PRINT@437,"   ":GOTO 200
500 A$="GARGLING.ARRGLARRGLARRGLARRGLARRGLARRGLARRGL"
510 GOSUB 1500
520 GOTO 200
550 A$="RUBBER-LIPS.BLBLBLBLBLBLBLBLBLBL"
560 GOSUB 1500
570 GOTO 200
600 A$="STEAM-TRAIN.SHSHSHSHSHSS,TOOT-TOOT,SHSHSH"
620 GOSUB 1500
640 GOTO 200
650 A$="SNEEZING.ASJ,ASJ,ASJ,ASJ-GZUNTITE"
660 GOSUB 1500
670 GOTO 200
700 A$="MACHINE-GUN.GGGGGGGGGGGGGGGGGGGGGGGGGGGGGG"
710 GOSUB 1500
720 GOTO 200
750 A$="WHISPERING.SHSH.SH..SH-SH-SH-SHSHSH-SH-SHSHSH"
760 GOSUB 1500
770 GOTO 200
800 A$="STARTER-MOTOR.IH-IH-IHIHIHIHIHIHIHIHIHIHIHIHIH"
810 GOSUB 1500
820 GOTO 200
850 A$="SIREN.OOWLOOWLOOWLOOWLOOWLOOWLOOWLOOOWLOOWL"
860 GOSUB 1500
870 GOTO 200
900 A$="DOGGY.WFFWFF-WFF-WFFWFF-RFF"
910 GOSUB 1500
920 A$="..THATS-PUHTHETIC"
930 GOSUB 1500
940 GOTO 200
950 INPUT" YOUR TRY ";A$
960 IF A$="",A$="YOU-DID-NOT-ENTER-ANY-THING.TRI--AGAN"
970 GOSUB 1500
980 '
990 RUN
1500 FOR T=1 TO LEN(A$):C=ASC(MID$(A$,T,1)):OUT12,C:NEXT
1520 LPRINT:LPRINT
1560 PRINT@437,"  ":A$="":RETURN
1600 END
```

```
1650 ' NO# 11 UPDATE DISK
1660 A$="DO-YOU-WAN-2-UP-DATE-THE-DISK":GOSUB 1500
1670 D$=INKEY$:D$=INKEY$
1680 IF INKEY$="N",220
1690 IF INKEY$="Y",1750
1700 IF INKEY$<>"Y" OR INKEY$<>"N",1670
1740 END
1750 ERA"TALK A":SAVE"TALK A":GOTO 220
2000 FOR L=31870 TO 32000
2020 PRINTL;PEEK(L);CHR$(PEEK(L))
2025 IF INKEY$=" ",2025
2030 NEXT
2040 LIST 20
```

INKEY$ INPUT ROUTINE BY PAUL LEON

As most programmers know the INPUT command on the VZ has it's limitations, but an INKEY$ INPUT routine can be tailored to your requirements. The routine below will not accept leading spaces, INVERSE or GRAPHIC characters. POKE30776,40 disables latter two inputs. Both ALPHA and NUMERICS are catered for. NC=VAL(IN$) in line 90 converts STRING to a NUMERIC value and ulike the INPUT command COMMA'S, etc are ignored. Imagine no more REDO's.

LE is the variable which sets the lenght of the INPUT. Only lines 200 to 310 are required, the rest are for demonstration purposes. Trailing spaces can be removed by line 300 or left in. This routine was used for most INPUTS in M/WORDS program on page 16.

```
10 '***********************************************************
12 '*  INKEY$ INPUT ROUTINE DESIGNED BY PAUL LEON FOR HIS DAD  *
14 '***********************************************************
18 :
20 CLS:POKE30744,96:COLOR,0:POKE30862,80:POKE30863,52
25 SP$="                              ":REM 30 SPACES
30 PRINT@39," INKEY$  INPUT ":PRINT
35 PRINT"  DEMONSTRATION   ROUTINE ":PRINT:PRINT
40 PRINT"  USE RUBOUT (;)  KEY TO ERASE":PRINT
45 PRINT"  PREVIOUS ENTERED  CHARACTERS":GOTO65
50 PRINT@385," LIKE TO TRY AGAIN - Y/N  ";:SOUND30,1:LE=1
55 GOSUB200:Y$=IN$:IFY$="Y"THEN65ELSEIFY$="N"THEN60ELSE50
60 CLS:END
65 PRINT@417,SP$:GOSUB105
70 PRINT@385," REMOVE BLANK SPACES - Y/N  ";:SOUND30,1:LE=1
75 GOSUB200:YN$=IN$
80 IFYN$="Y"THEN85ELSEIFYN$="N"THEN85ELSEGOSUB110:GOTO70
85 GOSUB110:PRINT@385," NO. CHAR'S FOR TEXT 1-16  ";:LE=2
90 GOSUB200:NC=VAL(IN$):IFNC<1ORNC>16THEN85ELSEPRINT@385,SP$
95 PRINT @385," ENTER TEXT  ";:SOUND30,1:LE=NC:GOSUB200
100 PRINT@417,SP$:PRINT@429,CHR$(34)IN$CHR$(34):GOTO50
105 PRINT@385,SP$:SOUND30,1:RETURN
110 PRINT@412,"    ":SOUND30,1:RETURN
180 :
190 REM" INKEY$ INPUT ROUTINE
200 CU$=CHR$(222)+CHR$(8):BS$=CHR$(8)+CHR$(127)+CU$:IN$=""
210 L=LEN(IN$):PRINT IN$;CU$;
220 A$=INKEY$:A$=INKEY$:IFA$=""THEN220ELSEX=USR(X)
230 POKE30776,40:IFA$=CHR$(13)THENPRINT" ":GOTO290
240 IFA$=";"ANDL>0THENPRINTBS$;:L=L-1:IN$=LEFT$(IN$,L)
250 IFL=LE THENSOUND20,1ELSEIFA$=" "ANDL=0THEN220
260 IFA$<" "ORA$>"^"ORA$=";"THEN220
270 IFL<LE THENPRINTA$;CU$;:IN$=IN$+A$:L=L+1
280 GOTO220
290 IFYN$="N"THENRETURN
300 IFRIGHT$(IN$,1)=" "THENIN$=LEFT$(IN$,LEN(IN$)-1):GOTO300
310 RETURN
```

USING A DISK TOKEN IN A BASIC PROGRAM :-

First we choose a disk token--133 will do. Then we insert this token in a brief BASIC program to test its operation. Enter the following program:-

```
10 LET:PRINT"TEST"
20 PRINT"AGAIN"
30 END
```

To replace the token for LET with our disk token,
POKE31469,133

If you now RUN you will get ERROR messages because your VZ's MICROPROCESSOR (the MP) picked up on token 133 at start of line 10, located the pointer for this token which directed it to address 31091 where it found and executed a JP301 instruction.

What we want to do is change the instruction at 31091 so that the MP will execute it, then return to line 10 and continue to run our BASIC program. The simplest change I know of is a RET instruction (Code 201). So, POKE31091,201 and RUN.

It works! The MP jumped to 31091 where the RETurn was encountered and executed and program RUN continued on to complete line 10 and then line 20. This tells us something important: when a token is encountered in a BASIC program, if a CALL instruction is executed, so that no matter where the MP may jump to run the M/L routine tied to that token, it can find its way back to its position in the current BASIC line if the M/L routine ends with a RET.

So let's try something a little more complicated. We'll set up a short M/L routine in a stretch of otherwise unused memory, in the COMMUNICATIONS REGION, that begins at address 31273 (LO=41; HI=122). First change the instruction at 31091 to JP 31273:

POKE31091,195:POKE31092,41:POKE 31093,122

Now add these lines to our BASIC program:

```
50 A=31273
60 INPUTB:POKEA,B:A=A+1:GOTO60
```

RUN50 and INPUT this sequence of numbers: 33,1,1,34,0,112,201

BREAK -- We have now set up a M/L routine at 31273 consisting of these three instructions:-

```
LD HL,257
LD(28672),HL
RET
```

The routine loads the H and L registers with '1' which is the POKE code for A, then POKEs these codes to the first two cells of screen memory (28672/3, top left corner) and RETurns.

So, now CLS, press <RETURN> and RUN.

?SYNTAX ERROR IN 10

But our M/L routine executed, did it not? There are two A's in top of screen. Only the MP returned to line 10 and BREAKed. Why? Because we changed the contents of the HL registers. When we use USR(x) to execute a M/L routine, USR saves the current contents of the HL registers on the STACK and when RET is encountered to bring the MP back to BASIC, the contents of the hl registers are restored. No matter how the registers may change in the course of executing the M/L routine, the MP continues execution of the BASIC program with the registers the same as when it began USR.

We must do the same with our M/L routine. The routine only uses the HL register pair so we need only save and restore these two registers. We do this by PUSHing HL on the STACK at the start of our routine and then POPing them from the STACK at the end of the routine.

```
229       PUSH HL
225       POP HL
```

Again RUN50 and INPUT this sequence of numbers: 229, 33, 1, 1, 34, 0, 112, 225, 201

BREAK. CLS, press <RETURN> and RUN.

This time our program ran through to END in line 30, executing our M/L routine, then returning to execute the two PRINT statements in lines 10 and 20.

Having successfully worked out a technique for making use of disk tokens, it only remains to design some interesting and useful M/L routines to execute with our new technique. That will be largely up to you. But here is an example:-

Video memory consists of 2048 bytes (from 28672 to 30719), of which only one quarter (512 bytes) is used for the screen display in text mode (28672 to 29183). We can use another block of 512 bytes of video memory as a video store, say from 29184 to 29695, and set up M/L routines to copy the text mode screen to our video store and recall the content of the video store back to screen. This can be done very easily entirely in BASIC, using PEEK and POKE in FOR NEXT loops. The advantage of M/L routines is that they are very fast, practically instantaneous.

Again RUN50 and INPUT the following sequence of numbers :-
229,33,0,112,17,0,114,1,0,2,237,176,225,201
229,33,0,114,17,0,112,1,0,2,237,176,225,201

And BREAK. Two routines have been set up in the COMMUNICATIONS REGION, a COPY (screen to video store) routine starting at 31273 and a RECALL (from video store to screen) routine starting at 31287.

We will use the disk token 165 to execute the COPY routine (PUT) and disk token 164 to execute the RECALL routine (GET). We must change the JP instructions in the COMMUNICATIONS REGION for these disk tokens to make them jumps to the start of our M/L routines:

```
POKE31107,41:POKE31108,122     POKE31104,55:POKE31105,122
```

We are now ready to use our new tokens and their M/L routines in BASIC programs. So let's try them out. First NEW your VZ.

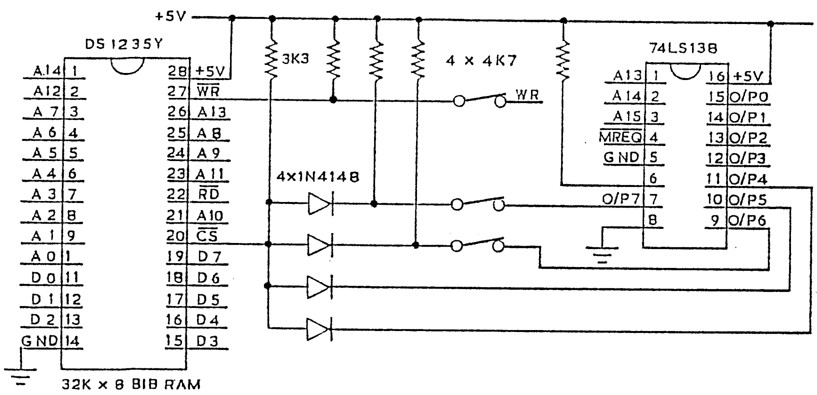
Enter this BASIC line :-
500 LET:RETURN
Now POKE31469,165 and enter this line :-
400 LET:RETURN
And POKE31469,164

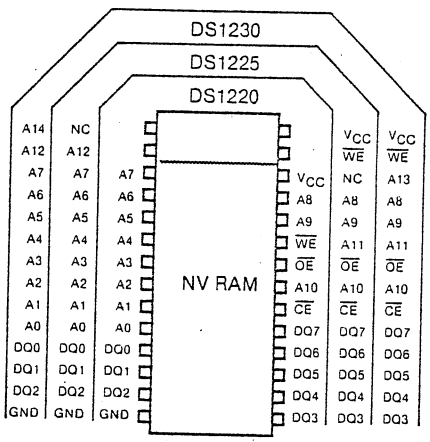
We have set up two BASIC subroutines, a PUT routine at line 500 and a GET routine at line 400 which can now be called by using GOSUBs to those lines from within our BASIC program or even called with GOSUBs in command mode.

Try it. Enter GOSUB500
NOW CLS and GOSUB400.

How about that eh? You can fill the screen with any data you please, then store it in the video store with a GOSUB500, and recall it to screen anytime you want with a GOSUB400.

32K BIB RAM BY JOE LEON . . . . .





This 32K BIB RAM is the last in the line of BIB RAMS and is included for completeness. At around $140.00 it is very expensive and 40% more than the VZ itself. I can't see too many persons rushing out to get one. A 16K BIB RAM would be more useful than above, but it's not available to my knowledge. And now to the circuit. As you can see there's not much to it. The 4 DIODES and 3K3 resistor form a 4 INPUT DIODE AND gate. A single 32K or two 16K decoded O/P's could have been used but were decided against for the following reasons.

If you have a W.P. cartidge then it wont work with circuit unless you disable top 16K. The 2 switches allow us to do so. Also if you use 32K BIB ram as an EPROM program developer then it's very desirable being able to switch out one or more 8K blocks. The switch on the WR (WRITE) line is there as a WRITE PROTECT switch when needed. The diagram on the right shows the pinouts of 2K, 8K and 32K BIB RAMS. If you have trouble locating a supplier for BIB RAMS then contact :-

NOVOCASTRIAN ELECTRONIC SUPPLIES PTY. LTD. (049) 62-1358 or (049) 62-2005
24 BROADMEADOW ROAD, BROADMEADOW, NSW, 2292, AUSTRALIA
MK48Z02 2K BIB RAM - $34.40 - Available X stock.
DS1220 2K BIB RAM - $ - ????
DS1225 8K BIB RAM - $45.00 - $54 - (150 or 200ns)
DS1235 32K BIB RAM - $140.00 - (150ns)

All DS (Dallas Semiconductor) BIB RAMS are available on indent only. Prices are aproximate, ckeck with supplier for correct price.

Please mention H.V.VZ.U.G. when making enquiries or purchasing any above. Ed.

The 8K BIB RAM is one of the more exiting projects that I built. My thanks to Dave Mitchell, Robert Quinn and Larry Taylor for their help with routines and information which I found most useful, thanks fellas.

Using the program I activated all the following extended basic commands :- RANDOM, DEFINT, DEFSNG, DEFDBL, RESUME, ON, SYSTEM, DELETE, AUTO, VARPTR, ERL, ERR, STRING$, MEM, FRE, POS, CINT, CSNG, CDBL and FIX.

The following disk words were also enabled with some of their names being changed. PUT, GET, CMD to DOS, OPEN to TRON, CLOSE to TROFF.

Now just by switching ROM 0 out and switching 8K BIB RAM in it's place all the Ext. Basic commands are at my disposal. Although routines for TRON and TROFF are in the ROM their place in the word table has been taken by other words. Renaming words wont activate routines, pointers have to be changed which is what I did.

As there are no routines in ROM for the disk commands CMD, PUT and GET other routines were used. R.Quinn's block move routine was placed in 2K BIB RAM at 6000H. As in his article PUT shifts the text screen up 512 bytes (1/2K) while GET brings it back to the text screen. PUT and GET gives me instant menu in my programs.

DOS :- I bet this command has you wondering. Quite often when I need to renumber part of a program I CLOAD W.Obrist's renumbering utility. When done I used to have to save program to tape as the renumbering utility clobbered the DOS. Now when I want to return to disk use I type in DOS and it reinitialises both the VZ and DOS. The basic program no longer responds to LIST or RUN, but by using Dave Mitchell's Ext. Dos OLD command the program is restored.

T - TEXT POKE :- This function is designed to poke TEXT to RAM. EG :- READY - How many times have you pressed RETURN over READY and got an ERROR message. By using TEXT POKE you can change it to REM as it produces no error message. The word READY lives at 6441-6446. Don't forget to put 2 spaces at end of REM as it is only 3 characters long.

P - POKE WORDS :- This is the option for enabling missing words. R.Quinn's WORD routine is used to identify missing words which are then typed in and as soon as RETURN is pressed they are POKED in. Refer to the two tables in pages 17-18. The left culumn shows all words while right column shows the gaps where the missing words go. Trailing spaces and inverse characters are disabled in this function so the correct number of characters have to be entered.

W - WORDS PEEK :- This is like above, but for viewing only.

M - MEM PEEK - This allows you to PEEK at both Hi and Lo Memory.

R - MOVE ROM X :- This options will move ROM 0, ROM 1 and DOS ROMS to C000-DFFF (49152-57343). This is where all the alterations are done and POKE WORDS routine only works in this area. MOVE ROM 0 before POKE.

So far in MEM PEEK HI and TEXT POKE HI routines, offsets for ROM 0 only (lines 300 and 510) were incorporated as the program was originally designed to enable missing words only and it sort of grew a bit. As the program is still under developement other options will be added like Lo-Byte/Hi-Byte POKE for use in changing word pointers, etc.

This 8K RAM with built in LITHIUM bateries has been very usefull in enabling missing words. Also as any one of the 3 ROMS can be put in 8K BIB RAM they can be altered, routines rewritten and just a few seconds later just by flicking a couple of switches Instant Eprom.

```
10 '***********************************************************
12 '*  M/WORDS - MISSING WORD ENABLING ROUTINE  BY JOE LEON  *
14 '*  WITH  ROUTINES  BY  ROBERT  QUINN   AND   PAUL  LEON  *
16 '***********************************************************
18 :
20 POKE30897,204:POKE30898,191:REM - SET NEW TOP OF MEMORY
25 CLS:GOTO100
30 REM"[illegible]
35 CLS:POKE30777,1:PRINTSS$;
40 D=127:FORR=5712TO A2 :P=PEEK(R):IFP>128THEND=D+1:PRINT
45 IFP>128THENPRINTUSING"####";R;D;:PRINT":";
50 IFP>128THENPRINTUSING"###";P;ELSEPRINTUSING"##";P;
55 IFP=129ANDD$="P"THENGOSUB400
60 IFP>0THENPRINTCHR$(P);" ";
65 IFP<192ANDP>159THENPRINT"WORD IS ";CHR$(P-128);
70 A$=INKEY$:A$=INKEY$:IFA$<>""THENSOUND20,1:C=NOTC
75 IFC=-1THEN70ELSEIFA$="Q"THEN80ELSENEXT
80 POKE30777,35:PRINT:PRINT:GOSUB1105:GOTO150
90 :
95 REM"[illegible]"
100 FOR BM=31273TO31286:READMB:POKEBM,MB:NEXT
105 DATA 229,33,0,0,17,0,192,1,0,32,237,176,225,201
110 :
150 CLEAR300:COLOR7:M$=" [illegible]   "
155 R$="[illegible]":SP$="       ":REM 7 SPACES
160 BL$="                                ":REM 32 SPACES
165 SS$="[illegible]":S3$="    "
170 EQ$=" ==============================":REM 30 = SYMBOLS
180 CLS:POKE30744,96:COLOR,0:POKE30862,80:POKE30863,52:GOTO700
190 :
200 PRINT@473,SP$;:PRINT@447,"  [illegible]   ";
205 SOUND30,1:LE=5:GOSUB2000:A1=VAL(IN$)
210 IFD$="T"THEN500
220 PRINT@473,SP$;:PRINT@447,"  [illegible]   ";
225 SOUND30,1:LE=5:GOSUB2000:A2=VAL(IN$)
230 IFD$="W"ORD$="P"THEN35ELSEIFA1=0THEN150
260 :
270 PRINT@448," [illegible]   ";:GOSUB915
275 LE=1:GOSUB2000:X$=IN$
280 IFX$="H"THEN300ELSEIFX$="L"THEN310ELSE270
300 A1=A1-16384:A2=A2-16384
310 CLS:POKE30777,1:PRINTSS$;
315 PRINT:FORL= A1 TO A2 :PRINTUSING" ######";L;
320 PRINTUSING" ###  ";PEEK(L);
325 PRINTCHR$(PEEK(L));
330 IFX$="H"ORP$="H"THENPRINTUSING"###### ";L+16384ELSEPRINT" "
335 IFINKEY$=" "THEN340ELSEIFINKEY$="Q"THEN345
340 IFINKEY$=""THEN335ELSENEXT
345 PRINT:PRINT@480,R$;:POKE30777,35:SOUND30,1
350 IFINKEY$=CHR$(13)THEN180ELSE350
390 :
400 PRINT:PRINT:PRINT"[illegible]";R;D :SOUND30,1
405 A3=R-16384
420 :
430 PRINTBL$;:PRINT"[illegible]    ";:POKE30777,35
435 SOUND30,1:LE=7:GOSUB2000:W$=IN$:IFW$=""THENRETURN
440 POKE A3,ASC(W$)+128:FOR I=1 TO LEN(W$)-1
445 POKE A3+I,ASC(MID$(W$,I+1,1)):NEXT:RETURN
500 PRINT@448," [illegible]   ";:GOSUB915
505 LE=1:GOSUB2000:P$=IN$
510 IFP$="H"THENA4=A1-16384:GOTO550ELSEIFP$="L"THENA4=A1:GOTO550
515 IFP$="Q"THEN180ELSE500
```

```
550 PRINT@448,BL$;:PRINT@449,"[illegible] ";:SOUND30,1
555 LE=16:GOSUB2000:T$=IN$
560 IFT$=""THEN700
565 FORI=0TOLEN(T$)-1:POKEA4+I,ASC(MID$(T$,I+1,1)):NEXT
570 A1=A4:A2=A4+LEN(T$)-1:GOTO310
590 :
600 PRINT@476,S3$
605 COLOR3:PRINT@447," [illegible]  ";:SOUND30,1
610 LE=1:GOSUB2000:RE$=IN$
615 IFRE$="0"THENLB=0:HB=0:GOTO635
620 IFRE$="1"THENLB=0:HB=32:GOTO635
625 IFRE$="D"THENLB=0:HB=64:GOTO635ELSEIFRE$="Q"THEN180ELSE605
635 POKE31275,LB:POKE31276,HB
640 POKE30862,41:POKE30863,122:PRINTUSR(0):GOTO180
690 :
700 CLS:POKE30777,1
705 PRINT" [illegible]":PRINTEQ$:PRINT
710 PRINT" [illegible] - TEXT  POKE  [illegible] - SAVE ROM 0":PRINT
715 PRINT" [illegible] - POKE WORDS  [illegible] - LOAD ROM 0":PRINT
720 PRINT" [illegible] - WORDS PEEK  [illegible] - MOVE ROM X":PRINT
725 PRINT" [illegible] - MEM   PEEK  [illegible] - REPL PROG.":PRINT
730 PRINT" [illegible] - DIRECTORY   [illegible] - SAVE PROG."
735 PRINT@416,EQ$;:POKE30777,35:GOSUB915
790 :
800 COLOR7:PRINT@448,M$;:LE=1:GOSUB2000:D$=IN$
805 IFD$="L",GOSUB900:GOTO1200ELSEIFD$="S",GOSUB900:GOTO1300
810 IFD$="$",GOSUB900:GOSUB1100:GOTO180
815 IFD$="&",GOSUB900:GOTO1000ELSEIFD$="^",GOSUB900:GOTO1000
820 IFD$="R"THEN600ELSEIFD$="P"THEN220ELSEIFD$="W"THEN220
825 IFD$="T"THEN200ELSEIFD$="M"THEN200ELSE800
890 :
900 COLOR3:PRINT@447," [illegible]";:GOSUB915
905 C$=INKEY$:C$=INKEY$:IFC$=""THEN905ELSEX=USR(X)
910 IFC$=" "THENRETURNELSEIFC$="Q"THEN180ELSE905
915 SOUND0,1;25,1;30,1:RETURN
990 :
1000 CLS:PRINT@480,"[illegible] T: M/WORDS";:SOUND30,1:ERA"M/WORDS"
1050 CLS:PRINT@480,"[illegible] T: M/WORDS ";:SOUND30,1:SAVE"M/WORDS"
1100 CLS:DIRA:STATUSA
1105 PRINT@480,R$;:GOSUB915
1110 IFINKEY$=CHR$(13)THEN180ELSE1110
1190 :
1200 CLS:PRINT@480,"[illegible] T: ROM-0PG";:SOUND30,1
1205 BLOAD"ROM-0PG":GOTO180
1300 CLS:PRINT@480,"[illegible] T: ROM-0PG";:SOUND30,1:ERA"ROM-0PG"
1400 CLS:PRINT@480,"[illegible] T: ROM-0PG ";:SOUND30,1
1405 BSAVE"ROM-0PG",C000,DFFF:GOTO180
1980 :
1990 REM"[illegible]
2000 CU$=CHR$(222)+CHR$(8):BS$=CHR$(8)+CHR$(127)+CU$:IN$=""
2005 L=LEN(IN$):PRINT IN$;CU$;
2010 A$=INKEY$:A$=INKEY$:IFA$=""THEN2010ELSEX=USR(X)
2015 POKE30776,40:IFA$=CHR$(13)THENPRINT" ":GOTO2045
2020 IFA$=";"ANDL>0THENPRINTBS$;:L=L-1:IN$=LEFT$(IN$,L)
2025 IFL=LE THENSOUND20,1ELSEIFA$=" "ANDL=0THEN2010
2030 IFA$<" "ORA$>"^"ORA$=";"THEN2010
2035 IFL<LE THENPRINTA$;CU$;:IN$=IN$+A$:L=L+1
2040 GOTO2010
2045 IFD$="T"ORD$="P"THENRETURN
2050 IFRIGHT$(IN$,1)=" "THENIN$=LEFT$(IN$,LEN(IN$)-1):GOTO2050
2055 RETURN
```

WORD ADD.	TOKEN	WORD	ROUTINE DEC	ADDRESS LO	HI	HEX
5712	128	END	7598	174	29	1DAE
5715	129	FOR	7329	161	28	1CA1
5718	130	RESET	312	56	1	0138
5723	131	SET	309	53	1	0135
5726	132	CLS	457	201	1	01C9
5729	133	CMD	31091	115	121	7973
5732	134	RANDOM	467	211	1	01D3
5738	135	NEXT	8886	182	34	22B6
5742	136	DATA	7941	5	31	1F05
5746	137	INPUT	8602	154	33	219A
5751	138	DIM	9736	8	38	2608
5754	139	READ	8687	239	33	21EF
5758	140	LET	7969	33	31	1F21
5761	141	GOTO	7874	194	30	1EC2
5765	142	RUN	7843	163	30	1EA3
5768	143	IF	8249	57	32	2039
5770	144	RESTORE	7569	145	29	1D91
5777	145	GOSUB	7857	177	30	1EB1
5782	146	RETURN	7902	222	30	1EDE
5788	147	REM	7943	7	31	1F07
5791	148	STOP	7593	169	29	1DA9
5795	149	ELSE	7943	7	31	1F07
5799	150	COPY	14610	18	57	3912
5803	151	COLOR	14493	157	56	389D
5808	152	VERIFY	14136	56	55	3738
5814	153	DEFINT	7683	3	30	1E03
5820	154	DEFSNG	7686	6	30	1E06
5826	155	DEFDBL	7689	9	30	1E09
5832	156	CRUN	14126	46	55	372E
5836	157	MODE	11875	99	46	2E63
5840	158	SOUND	11253	245	43	2BF5
5845	159	RESUME	8111	175	31	1FAF
5851	160	OUT	11003	251	42	2AFB
5854	161	ON	8044	108	31	1F6C
5856	162	OPEN	31097	121	121	7979
5860	163	FIELD	31100	124	121	797C
5865	164	GET	31103	127	121	797F
5868	165	PUT	31106	130	121	7982
5871	166	CLOSE	31109	133	121	7985
5876	167	LOAD	31112	136	121	7988
5880	168	MERGE	31115	139	121	798B
5885	169	NAME	31118	142	121	798E
5889	170	KILL	31121	145	121	7991
5893	171	LSET	31127	151	121	7997
5897	172	RSET	31130	154	121	799A
5901	173	SAVE	31136	160	121	79A0
5905	174	SYSTEM	0	0	0	0000
5911	175	LPRINT	8295	103	32	2067
5917	176	DEF	31067	91	121	795B
5920	177	POKE	11441	177	44	2CB1
5924	178	PRINT	8303	111	32	206F
5929	179	CONT	7652	228	29	1DE4
5933	180	LIST	11054	46	43	2B2E
5937	181	LLIST	11049	41	43	2B29
5942	182	DELETE	11206	198	43	2BC6
5948	183	AUTO	8200	8	32	2008
5952	184	CLEAR	7802	122	30	1E7A
5957	185	CLOAD	13910	86	54	3656
5962	186	CSAVE	13481	169	52	34A9
5967	187	NEW	6985	73	27	1B49
5970	188	TAB(	NO ADDRESS LIST			
5974	189	TO	NO ADDRESS LIST			

WORD ADD.	TOKEN	WORD IN CHR$ AND ASCII
5712	128:197■	78N 68D
5715	129:198■	79O 82R
5718	130:210■	69E 83S 69E 84T
5723	131:211■	69E 84T
5726	132:195■	76L 83S
5729	133:129■	0 0
5732	134:129■	0 0 0 0 0
5738	135:206■	69E 88X 84T
5742	136:196■	65A 84T 65A
5746	137:201■	78N 80P 85U 84T
5751	138:196■	73I 77M
5754	139:210■	69E 65A 68D
5758	140:204■	69E 84T
5761	141:199■	79O 84T 79O
5765	142:210■	85U 78N
5768	143:201■	70F
5770	144:210■	69E 83S 84T 79O 82R 69E
5777	145:199■	79O 83S 85U 66B
5782	146:210■	69E 84T 85U 82R 78N
5788	147:210■	69E 77M
5791	148:211■	84T 79O 80P
5795	149:197■	76L 83S 69E
5799	150:195■	79O 80P 89Y
5803	151:195■	79O 76L 79O 82R
5808	152:214■	69E 82R 73I 70F 89Y
5814	153:129■	0 0 0 0 0
5820	154:129■	0 0 0 0 0
5826	155:129■	0 0 0 0 0
5832	156:195■	82R 85U 78N
5836	157:205■	79O 68D 69E
5840	158:211■	79O 85U 78N 68D
5845	159:129■	0 0 0 0 0
5851	160:207■	85U 84T
5854	161:129■	0
5856	162:129■	0 0 0
5860	163:129■	0 0 0 0
5865	164:129■	0 0
5868	165:129■	0 0
5871	166:129■	0 0 0 0
5876	167:129■	0 0 0
5880	168:129■	0 0 0 0
5885	169:129■	0 0 0
5889	170:129■	0 0 0
5893	171:129■	0 0 0
5897	172:129■	0 0 0
5901	173:129■	0 0 0
5905	174:129■	0 0 0 0 0
5911	175:204■	80P 82R 73I 78N 84T
5917	176:129■	0 0
5920	177:208■	79O 75K 69E
5924	178:208■	82R 73I 78N 84T
5929	179:195■	79O 78N 84T
5933	180:204■	73I 83S 84T
5937	181:204■	76L 73I 83S 84T
5942	182:129■	0 0 0 0 0
5948	183:129■	0 0 0
5952	184:195■	76L 69E 65A 82R
5957	185:195■	76L 79O 65A 68D
5962	186:195■	83S 65A 86V 69E
5967	187:206■	69E 87W
5970	188:212■	65A 66B 40(
5974	189:212■	79O

WORD ADD.	TOKEN	WORD	ROUTINE DEC	ADDRESS LO	HI	HEX
5976	190	FN	NO ADDRESS LIST			
5978	191	USING	NO ADDRESS LIST			
5983	192	VARPTR	NO ADDRESS LIST			
5989	193	USR	NO ADDRESS LIST			
5992	194	ERL	NO ADDRESS LIST			
5995	195	ERR	NO ADDRESS LIST			
5998	196	STRING$	NO ADDRESS LIST			
6005	197	INSTR	NO ADDRESS LIST			
6010	198	POINT	NO ADDRESS LIST			
6015	199	TIME$	NO ADDRESS LIST			
6020	200	MEM	NO ADDRESS LIST			
6023	201	INKEY$	NO ADDRESS LIST			
6029	202	THEN	NO ADDRESS LIST			
6033	203	NOT	NO ADDRESS LIST			
6036	204	STEP	NO ADDRESS LIST			
6040	205	+	NO ADDRESS LIST			
6041	206	-	NO ADDRESS LIST			
6042	207	*	NO ADDRESS LIST			
6043	208	/	NO ADDRESS LIST			
6044	209	^	NO ADDRESS LIST			
6045	210	AND	NO ADDRESS LIST			
6048	211	OR	NO ADDRESS LIST			
6050	212	>	NO ADDRESS LIST			
6051	213	=	NO ADDRESS LIST			
6052	214	<	NO ADDRESS LIST			
6053	215	SGN	2442	138	9	098A
6056	216	INT	2871	55	11	0B37
6059	217	ABS	2423	119	9	0977
6062	218	FRE	10196	212	39	27D4
6065	219	INP	10991	239	42	2AEF
6068	220	POS	10229	245	39	27F5
6071	221	SQR	5095	231	19	13E7
6074	222	RND	5321	201	20	14C9
6077	223	LOG	2057	9	8	0809
6080	224	EXP	5177	57	20	1439
6083	225	COS	5441	65	21	1541
6086	226	SIN	5447	71	21	1547
6089	227	TAN	5544	168	21	15A8
6092	228	ATN	5565	189	21	15BD
6095	229	PEEK	11434	170	44	2CAA
6099	230	CVI	31058	82	121	7952
6102	231	CVS	31064	88	121	7958
6105	232	CVD	31070	94	121	795E
6108	233	EOF	31073	97	121	7961
6111	234	LOC	31076	100	121	7964
6114	235	LOF	31079	103	121	7967
6117	236	MKI$	31082	106	121	796A
6121	237	MKS$	31085	109	121	796D
6125	238	MKD$	31088	112	121	7970
6129	239	CINT	2687	127	10	0A7F
6133	240	CSNG	2737	177	10	0AB1
6137	241	CDBL	2779	219	10	0ADB
6141	242	FIX	2854	38	11	0B26
6144	243	LEN	10755	3	42	2A03
6147	244	STR$	10294	54	40	2836
6151	245	VAL	10949	197	42	2AC5
6154	246	ASC	10767	15	42	2A0F
6157	247	CHR$	10783	31	42	2A1F
6161	248	LEFT$	10849	97	42	2A61
6166	249	RIGHT$	10897	145	42	2A91
6172	250	MID$	10906	154	42	2A9A

WORD ADD.	TOKEN	WORD IN CHR$ AND ASCII
5976	190:129	0
5978	191:213	83S 73I 78N 71G
5983	192:129	0 0 0 0 0
5989	193:213	83S 82R
5992	194:129	0 0
5995	195:129	0 0
5998	196:129	0 0 0 0 0 0
6005	197:129	0 0 0 0
6010	198:208	79O 73I 78N 84T
6015	199:129	0 0 0 0
6020	200:129	0 0
6023	201:201	78N 75K 69E 89Y 36$
6029	202:212	72H 69E 78N
6033	203:206	79O 84T
6036	204:211	84T 69E 80P
6040	205:171	WORD IS +
6041	206:173	WORD IS -
6042	207:170	WORD IS *
6043	208:175	WORD IS /
6044	209:222	
6045	210:193	78N 68D
6048	211:207	82R
6050	212:190	WORD IS >
6051	213:189	WORD IS =
6052	214:188	WORD IS <
6053	215:211	71G 78N
6056	216:201	78N 84T
6059	217:193	66B 83S
6062	218:129	0 0
6065	219:201	78N 80P
6068	220:129	0 0
6071	221:211	81Q 82R
6074	222:210	78N 68D
6077	223:204	79O 71G
6080	224:197	88X 80P
6083	225:195	79O 83S
6086	226:211	73I 78N
6089	227:212	65A 78N
6092	228:193	84T 78N
6095	229:208	69E 69E 75K
6099	230:129	0 0
6102	231:129	0 0
6105	232:129	0 0
6108	233:129	0 0
6111	234:129	0 0
6114	235:129	0 0
6117	236:129	0 0 0
6121	237:129	0 0 0
6125	238:129	0 0 0
6129	239:129	0 0 0
6133	240:129	0 0 0
6137	241:129	0 0 0
6141	242:129	0 0
6144	243:204	69E 78N
6147	244:211	84T 82R 36$
6151	245:214	65A 76L
6154	246:193	83S 67C
6157	247:195	72H 82R 36$
6161	248:204	69E 70F 84T 36$
6166	249:210	73I 71G 72H 84T 36$
6172	250:205	73I 68D 36$

FOR SALE - EXTENDED DOS V1.0

EXTENDED DOS VERSION 1.0 (C) COMMANDS :-

MERGE - MERGES basic file from disk with program in memory.
DIRA - See example - T:MENU B:PATCH3.1 B:WORDPROC
B:EXTDOS E B:EXTDOS R W:DOS-INST
LDIRA - As above, but to screen and printer.
DIRB - See example - T:MENU 01 00 7AE9 801B 0532
B:PATCH3.1 01 0B 7200 771F 051F
LDIRB - As above, but to screen and printer.
STATUSA - Prints free disk space to screen on one line.
LSTATUSA - As above, but to screen and printer, see below.
534 RECORDS FREE 63.500K FREE
OLD - Restores a program after using the NEW command.
OLD. - Prints START, END and LENGHT of program in memory in HEX.
DEC XXXXX - Converts DECIMAL to HEX
HEX XXXX - Converts HEX to DECIMAL
STATUSA and LSTATUSA also works with Version 1.0 DOS.

The EXTENDED DOS is available in the two versions below :-
EXTDOS R - T.O.M. SEEKING (SELF RELOCATING)
EXTDOS E - FOR 2K RAM AT 6000-67FF HEX
Price - $10.00 each or the two for $15.00. Availble from :-

Dave MITCHELL - (079) 27 8519
24 ELPHINSTONE STREET NORTH ROCKHAMPTON QUEENSLAND 4701

FOR INFORMATION IN NEWCASTLE AREA :- Joe LEON - (049) 51 2756

FOR SALE - DATABASE - DISK / TAPE

DATA - 16k - VZ DATABASE. Enter data into records thirty characters long (accepts graphic characters). Runs on VZ 200+16k or VZ 300. Available on disk as DISK DATABASE or on tape as CASSETTE DATABASE.

Facilities include data entry into record of choice, into last record chosen, next record, auto-next for fast data entry, edit keys so you don't have to re-enter entire content of a record, delete a record, delete a block of records, gap delete, insert, gap insert, fast alphabetical sort of records--start anywhere in records ; number sort ; swap any two records ; page display--ten records per page ; display current page, next page, previous page, flip backward and forward through datafile, swap any two pages, fast search of entire datafile for a sequence of characters--anywhere in records, hardcopy your records--especially suited for VZ printer plotter ; menu etc.

Disk DATA has Directory and ERASE commands, saves a datafile or any part thereof as a single binary file which loads back quickly. Cassette DATA CSAVES a datafile as a single T file--no slow loading of multitudes of D files! All instructions for using DATA are stored on disk and tape as datafiles--run DATA, load an instruction file and page through it. This program certainly stands out amongst the crowd of other such programs of it's type.

PRICE - $20.00 for DISK or CASSETTE DATABASE - Please make all Cheques and Money Orders payable to and is available from :-
SCOTT LE BRUN 5 CAMERON COURT WANTIRNA VIC. 3152